Computing 29, 175—181 (1982)

Computing
© by Springer-Verlag 1982

Algorithm / Algorithmus 48

A Fast Algorithm for Clusterwise Linear Regression

H. Späth, Oldenburg

Received September 23, 1981

Abstract — Zusammenfassung

Algorithm 48. A Fast Algorithm for Clusterwise Linear Regression. A fast implementation of a formerly [5] published algorithm is given.

AMS Subject Classifications: 62J05, 62H30, 65F20, 65D10.

Key words: Cluster analysis, linear regression.

Algorithmus 48. Ein schneller Algorithmus zur klassenweise linearen Regression. Für einen früher publizierten Algorithmus [5] wird eine schnelle Implementation angegeben.

1. Problem and Purpose

Let be given m observations (y_i, a_{ik}) $(i=1,\ldots,m,\ k=1,\ldots,l)$ with $m>l$. Then this algorithm tries to find a partition $C_1,\ldots,C_n$ of given length n for these observations, i.e. $C_j \subset M=\{1,\ldots,m\}$, $|C_j|\geq l$, $C_j \cap C_k=\emptyset$ for $j\neq k$, $C_1\cup\ldots\cup C_n=M$, and regression coefficients $x^{(j)}=(x_1^{(j)},\ldots,x_l^{(j)})$ $(j=1,\ldots,n)$ such that

$$\sum_{j=1}^{n} \min_{x^{(j)}} \sum_{i\in C_j} \left(y_i - \sum_{k=1}^{l} a_{ik} x_k^{(j)}\right)^2 \rightarrow \min.$$

This objective is reasonable when the number m of observations is relatively large as against to the number l of variables and/or when the observations might stem from different groups. For $l=1$ and $a_{i1}=1$ $(i=1,\ldots m)$ you will have the well-known minimum variance criterion from cluster analysis [6] in one dimension.

The above idea was considered in [5] and an inefficient program was given, too, that additionally had a small mistake [7] but could easily be extended for L_p norms. The purpose of the present paper is to give a really efficient implementation of precisely the same algorithm.

2. Numerical Method

For larger values of m an exact optimum of the objective function cannot be found within reasonable computing times [6]. The following heuristic method is nearly identical to that exchange method that is successfully used for the minimum variance criterion and the quadratic assignment problem.

Step 1: Choose some initial partition $C_1, \ldots, C_n$ that is feasible, i.e. $|C_j| \geq l$, and some starting observation $i = i_a$.

Step 2: Set $i := i+1$ and reset $i := 1$ if $i > m$. For $i \in C_j$ and $|C_j| > l_n$ $(l_n \geq l)$ examine whether there are clusters C_p with $p \neq j$ such that shifting the observation i from C_j to C_p reduces the objective function. If so, then choose C_r such that the reduction becomes maximal and redefine $C_j := C_j - \{i\}$, $C_r = C_r \cup \{i\}$. Otherwise return to step 2.

Step 3: Repeat step 2 as long as you get any reduction, i.e. as long as i has been increased m times without any change.

This stepwise optimal method works sequentially on the observations. Its result depends on the initial partition, on the starting observation i_a and on the choice of l_n. Normally, in order to get a suitable approximation for an optimal solution, it is sufficient to try several possibilities and to select the best final partition. For instance you can use the standard initial partition $C_j = \{i : i \in M, i \equiv j \pmod n\}$, several values for i_a, and l_n.

3. Implementation

A very inefficient but numerically stable implementation of this algorithm was given in [5]. Using suitable up- and downdating processes for the regressions, see e.g. [1], [2], [4], results in a far more efficient algorithm. Here we have decided to use slightly modified FORTRAN versions of the ALGOL procedures *include* and *regress* from [3] working with Givens' rotations. As the downdating process may become numerically instable for a resulting small number of observations, see [3], it is recommended to use $l_n \gg l$, say $l_n \approx 2l$, if m/n is large enough. As a precaution all internal operations are done in double precision. In order to detect numerical instabilities it is recommended to restart the subroutine with the found final partition and to compare the results.

4. FORTRAN Subroutine

The formal parameters of the following subroutine CWDLRS are precisely explained within the comment cards at the beginning of the listing. As against to CWDLR from [5] CWDLRS is self-contained, internally uses double precision and parameter communication with the central auxiliary subroutine INEXCL is done via COMMON statements that is faster than passing parameters via argument lists. Thus the two subroutines cannot directly be compared.

```
      SUBROUTINE CWDLRS (M,L,LDIM,N,A,Y,P,KP,IA,LN,Q,X,E,ES)
C
C  THE PARAMETERS ARE DEFINED AS FOLLOWS:
C
C  M          NUMBER OF OBSERVATIONS
C
C  L          NUMBER OF INDEPENDENT VARIABLES
C
C  LDIM       FIRST DIMENSION OF A AND X (BELOW) IN CALLING PROGRAM
C
C  N          NUMBER OF DESIRED CLUSTERS   (1 <= N <= M/LN)
C
C  A(LDIM,M)  THE ARRAY A HAS TO CONTAIN THE GIVEN (M,L)-MATRIX OF
C             OBSERVATIONS FOR THE INDEPENDENT VARIABLES
```

```
C
C  Y(M)        THE ARRAY Y HAS TO CONTAIN  THE GIVEN M-VECTOR OF
C              VALUES FOR THE DEPENDENT VARIABLE
C
C  P(M)        FOR KP.NE.0 THIS INTEGER M-VECTOR INITIALLY HAS TO
C              CONTAIN A FEASIBLE PARTITION OF LENGTH N VIA P(I)=J
C              (I=1,...,M, J=1,...,N).
C              FOR KP.EQ.0 THE STANDARD INITIAL PARTITION IS GENERATED.
C              ON OUTPUT P WILL CONTAIN THE FINAL PARTITION OBTAINED
C              BY THE EXCHANGE METHOD
C
C  KP          SEE P ABOVE
C
C  IA          THE EXCHANGE METHOD IS STARTED WITH OBSERVATION
C              NUMBER IA + 1 (MOD M). NORMALLY ONE SETS IA=0.
C              CHANGING IA AND USING THE SAME PARTITION P GIVES ANOTHER
C              (BETTER, EQUAL, OR WORSE) VALUE FOR THE OBJECTIVE
C              FUNCTION
C
C  LN          THE MINIMUM NUMBER OF OBSERVATIONS DESIRED IN EACH
C              CLUSTER. WE MUST HAVE AT LEAST LN >= L. FOR LN <= 0
C              THIS VALUE IS AUTOMATICALLY GENERATED. IF THERE ARE
C              ENOUGH OBSERVATIONS IN RELATION TO THE NUMBER OF
C              CLUSTERS IT IS RECOMMENDED (ALSO FOR IMPROVING
C              NUMERICAL STABILITY) TO USE LN >> L
C
C  Q(N)        THE J-TH COMPONENT OF THIS INTEGER VECTOR WILL CONTAIN
C              THE NUMBER OF OBSERVATIONS IN THE J-TH CLUSTER
C
C  X(LDIM,N)   WILL CONTAIN THE (N,L)-MATRIX OF SOLUTION PARAMETERS,
C              I.E. X(K,J) (K=1,...,L) ARE THE REGRESSION COEFFICIENTS
C              FOR THE J-TH CLUSTER OF OBSERVATIONS (J=1,...,N)
C
C  E(N)        THE J-TH COMPONENT WILL CONTAIN THE ERROR SUM OF
C              SQUARES FOR THE J-TH CLUSTER
C
C  ES          WILL CONTAIN THE SUM OF THE E(J)
C
C
C  OTHER ARRAYS ARE FOR WORKING SPACE. COMMUNICATION WITH THE
C  SUBROUTINE INEXCL IS DONE VIA THE LABELED COMMON /INEX/.
C  FOR L > 10 AND/OR N > 20 DIMENSIONS HAVE TO BE ADAPTED IN
C  CWDLRS AND INEXCL. THE RIGHT NUMBERS ARE INDICATED IN THE
C  FOLLOWING C-CARDS
C
      INTEGER P,Q,U,V,W,PI,UI
      DIMENSION A(LDIM,M),Y(M),X(LDIM,N),E(N),P(M),Q(N)
C
C     LDIM <= 10,  L <= LDIM,  N <= 20
C
C     NR= ((L-1)*L)/2
C
C     DIMENSION ED(N),XI(L),D(L,N),T(L,N),R(NR,N),DC(L),TC(L),
C    *          RC(NR),DA(L),TA(L),RA(NR),DB(L),TB(L),RB(NR)
C
      REAL*8 ED(20),XI(10),D(10,20),T(10,20),R(45,10),DA(10),TA(10),
     *       RA(45),DB(10),TB(10),RB(45),DC(10),TC(10),RC(45),
     *       YI,WI,SS,DSUM,SA,SB,SF,FF,EA,ZERO,ONE,BIG
C
      COMMON /INEX/ XI,D,T,R,DC,TC,RC,YI,WI,SS,ZERO,ONE,LL,L2,K,LU,NRV
C
C     BIG  LARGEST NUMBER ON YOUR COMPUTER
C
      BIG=1.D50
C
      ZERO=0.D0
      ONE=1.D0
C
      LL=L
      L1=LL+1
      L2=LL+LL
      IF (LN.LE.0) LN=LL
      NR=((LL-1)*LL)/2
C
```

```
C     GENERATION OF INITIAL PARTITION IF DESIRED
C
      IF(KP.NE.0) GOTO 2
      K=0
      DO 1 I=1,M
           K=K+1
           IF(K.GT.N) K=K-N
           P(I)=K
    1 CONTINUE
C
C     INITIALIZATION TO ZERO
C
    2 DSUM=ZERO
      DO 5 K=1,N
           Q(K)=0
           ED(K)=ZERO
           DO 3 U=1,L
                D(U,K)=ZERO
                T(U,K)=ZERO
    3      CONTINUE
           DO 4 V=1,NR
                R(V,K)=ZERO
    4      CONTINUE
    5 CONTINUE
C
C     UPDATE FOR INITIAL PARTITION
C
      DO 9 I=1,M
           K=P(I)
           IF(KP.NE.0.AND.K.LT.0.OR.K.GT.N) RETURN
           Q(K)=Q(K)+1
           WI=ONE
           YI=Y(I)
           DO 6 U=1,L
                XI(U)=A(U,I)
    6      CONTINUE
           SS=ED(K)
           CALL INEXCL
           ED(K)=SS
           DO 7 U=1,LU
                D(U,K)=DC(U)
                T(U,K)=TC(U)
    7      CONTINUE
           DO 8 V=1,NRV
                R(V,K)=RC(V)
    8      CONTINUE
    9 CONTINUE
      DO 10 K=1,N
            IF(Q(K).LT.LN) RETURN
            DSUM=DSUM+ED(K)
   10 CONTINUE
      IF(N.EQ.1) GOTO 22
C
C     START OF THE EXCHANGE METHOD
C
      IS=IA
      IT=0
   11 IS=IS+1
      IF(IS.GT.M) IS=IS-M
      IF(IT.EQ.M) GOTO 22
      J=P(IS)
C
C     IF THE NUMBER OF ELEMENTS OF THIS CLUSTER IS TOO SMALL
C     THEN DO NOT REMOVE THE OBSERVATION
C
      IF(Q(J).LE.LN) GOTO 11
      SF=BIG
      DO 18 K=1,N
            SS=ED(K)
            YI=Y(IS)
            DO 12 U=1,L
                  XI(U)=A(U,IS)
   12        CONTINUE
             WI=ONE
             IF(K.EQ.J) WI= - ONE
             CALL INEXCL
             IF(K.NE.J) GOTO 15
             SA=SS
             DO 13 U=1,L
                   DA(U)=DC(U)
                   TA(U)=TC(U)
   13        CONTINUE
             DO 14 V=1,NR
                   RA(V)=RC(V)
   14        CONTINUE
             GOTO 18
   15        SB=SS
             FF=SB-ED(K)
             IF(FF.GT.SF) GOTO 18
             SF=FF
             W=K
             DO 16 U=1,L
                   DB(U)=DC(U)
                   TB(U)=TC(U)
   16        CONTINUE
             DO 17 V=1,NR
                   RB(V)=RC(V)
   17        CONTINUE
   18  CONTINUE
       EA=ED(J)-SA
       IF(SF.LT.EA) GOTO 19
C
C      DO NOT EXCHANGE
C
       IT=IT+1
       GOTO 11
   19  IT=0
C
C      DO EXCHANGE
C
       P(IS)=W
       Q(J)=Q(J)-1
       Q(W)=Q(W)+1
       ED(J)=SA
       ED(W)=ED(W)+SF
       DSUM=DSUM+SF-EA
       DO 20 U=1,L
             D(U,J)=DA(U)
             D(U,W)=DB(U)
             T(U,J)=TA(U)
             T(U,W)=TB(U)
   20  CONTINUE
       DO 21 V=1,NR
             R(V,J)=RA(V)
             R(V,W)=RB(V)
   21  CONTINUE
       GOTO 11
C
C      CALCULATION OF REGRESSION COEFFICIENTS FOR FINAL PARTITION
C
   22  DO 26 K=1,N
       E(K)=ED(K)
       DO 25 W=1,L
             U=L1-W
             TA(U)=T(U,K)
             IF(U.EQ.L) GOTO 24
             NR=((U-1)*(L2-U))/2+1
             U1=U+1
             DO 23 V=U1,L
                   TA(U)=TA(U)-R(NR,K)*TA(V)
                   NR=NR+1
   23        CONTINUE
   24        X(U,K)=TA(U)
   25  CONTINUE
   26  CONTINUE
       ES=DSUM
       RETURN
       END
```

```
      SUBROUTINE INEXCL
      INTEGER U,U1,V
      REAL*8 XI(10),D(10,20),T(10,20),R(45,10),DC(10),TC(10),
     *       RC(45),YI,WI,SS,ZERO,ONE,
     *       XU,DU,WIXU,DP,HU,CB,SB,XV,RN,TN
      COMMON /INEX/ XI,D,T,R,DC,TC,RC,YI,WI,SS,ZERO,ONE,LL,L2,K,LU,NRV
      DO 3 U=1,LL
           IF(WI.EQ.ZERO) GOTO 4
           XU=XI(U)
           IF(XU.EQ.ZERO) GOTO 3
           DU=D(U,K)
           WIXU=WI*XU
           DP=DU+WIXU*XU
           HU=ONE/DP
           CB=DU*HU
           SB=WIXU*HU
           WI=WI*CB
           DC(U)=DP
           IF(U.EQ.LL) GOTO 2
           U1=U+1
           NR=((U-1)*(L2-U))/2+1
           DO 1 V=U1,LL
                XV=XI(V)
                RN=R(NR,K)
                XI(V)=XV-XU*RN
                RC(NR)=CB*RN+SB*XV
                NRV=NR
                NR=NR+1
    1      CONTINUE
    2      XV=YI
           TN=T(U,K)
           YI=XV-XU*TN
           TC(U)=CB*TN+SB*XV
           LU=U
    3 CONTINUE
      SS=SS+WI*YI*YI
    4 RETURN
      END
```

5. Computing Time

As the computing time heavily depends on the initial partition, on i_a and on l_n, it is not possible to give a precise estimate. Normally about six passes through the observations are enough. For an example with $m=96$, $l=5$, $l_n=l$, $n=2,3,4$, $i_a=0$, $KP=0$ (standard initial partition) CWDLRS has needed about 30 seconds on a TR 440 computer (about half as fast as an IBM 370/158). In this case CWDLRS was about 20 times faster than the corrected version of CWDLR, see [5], [7]. The gain will be larger for higher values of l, n, and m.

References

[1] Daniel, J. W., Gragg, W. B., Kaufman, L., Stewart, G. W.: Reorthogonalization and stable algorithms for updating the Gram-Schmidt QR factorization. Math. Comp. *30*, 722–795 (1976).

[2] Gentleman, M. W.: Least squares computations by Givens transformations without square roots. J. Inst. Maths Applics *12*, 329–336 (1973).

[3] Gentleman, M. W.: Algorithm AS 75. Basic procedures for large, sparse or weighted least squares problems. Appl. Statist. *23*, 448–454 (1974).

[4] Gragg, W. B., Leveque, R. J., Trangenstein, J. A.: Numerically stable methods for updating regressions. J. Am. Statist. Ass. *74*, 161–168 (1979).

[5] Späth, H.: Algorithm 39. Clusterwise linear regression. Computing *22*, 367–373 (1979).
[6] Späth, H.: Cluster analysis algorithms for data reduction and classification of objects. Chichester 1980.
[7] Späth, H.: Correction to Algorithm 39. Clusterwise linear regression. Computing *26*, 275 (1981).

Prof. Dr. H. Späth
Fachbereich Mathematik/Informatik
Universität Oldenburg
Postfach 2503
D-2900 Oldenburg
Federal Republic of Germany

Verleger: Springer-Verlag, Mölkerbastei 5, A-1010 Wien. — Herausgeber: Prof. Dr. Hans J. Stetter, Institut für Numerische Mathematik der Technischen Universität Wien, Gußhausstraße 27—29, A-1040 Wien. — Redaktion: Gußhausstraße 27—29, A-1040 Wien. — Hersteller: Satz Austro-Filmsatz Richard Gerin, Fischergasse 1, A-1020 Wien, Druck Paul Gerin, Zirkusgasse 13, A-1021 Wien. — Verlagsort: Wien. — Herstellungsort: Wien. — Printed in Austria.